18. Compare Bayes Classifier and KNN.

- **Step 1: Study [Color Classification using Bayes Theorem]**
- **Step 2: Apply [Bayes Theorem] to re-calculate the [sample data] (assuming K=8)**
- **Step 3: Compare the result of [Bayes Classifier] and [KNN.]**

Calculate KNN for Sample Data-

- To locate the neighbors for a new piece of data within a dataset we must first calculate the Euclidean distance between each record in the dataset to the new piece of data.

Euclidean Distance = sqrt(sum i to N (x1_i – x2_i)^2)

- Once distances are calculated, we must sort all of the records in the training dataset by their distance to the new data. We can then select the top k to return as the most similar neighbors.
- Since K = 8, we select the 8 closest neighbors.

X1	X2	Y	Distance to each neighbor = (TargetX1-DataX1)^2 +(TargetX2-DataX2)^2 = (6-X1)^2+(5-X2)^2	K =Number of nearest neighbors =8
4	3	+	8	
1	3	+	29	
3	3	+	13	
3	7	+	13	
7	4	+	2	+
4	1	+	20	
6	5	+	0	+
5	6	+	2	+
3	7	+	13	
6	2	+	9	
4	6	-	5	
4	4	-	5	
5	8	-	10	
7	8	-	10	
5	6	-	2	-
10	5	-	16	
7	6	-	2	-
4	10	-	29	

9	**7**	**-**	**13**	
5	**4**	**-**	**2**	-
8	**5**	**-**	**4**	
6	**6**	**-**	**1**	-
7	**4**	**-**	**2**	-
8	**8**	**-**	**13**	
6	**5**	**?(-)**		-

Baye's Theorem to re-calculate the sample data -

Prior probability for "+" ==> P(+)

= Number of "+" objects / Total number of objects = 10/24 = 0.416

Prior probability for "-" ==> P(-)

= Number of "-" objects / Total number of objects = 14/24 = 0.58

- Since K= 8, Check the Probability of X fall in **"+"** or **"–".**

Likelihood of X given **"+"**	Likelihood of X given "-"
Probability of X given "+"	Probability of X given "-"
P (X | +)	P (X | -)
count (+, X) / count (+)	count (-, X) / count (-)
Number of **"+"** in the vicinity (nearest neighbors) of X ------------------------------------- Total number of **"+"** cases	Number of **"-"** in the vicinity (nearest neighbors) of X ------------------------------------- Total number of **"-"** cases
3/10	5/14

Applying Bayes' Theorem

Posterior probability of X being **"+"**	Posterior probability of X being "-"
P **(+** | X)	P (- | X)
Prior probability of + * Likelihood of X given + / Prior probability of X	Prior probability of - * Likelihood of X given - / Prior probability of X
P(+) * P(X | +) / P(X)	P(-) * P(X | -) / P(X)
P (+) * P (X | +)	P (-) * P (X | -)
(10/24) * (3/10)	(14/2 4) * (5/14)

0.125	0.208 **i.e., X is "-"**

Comparison -

- The result of Bayesian inference depends strongly on prior probabilities, which must be available in order to apply the method directly. Since the Bayes theorem provides a principled way to calculate the posterior probability of each hypothesis given the training data, we can use it as the basis for straightforward learning algorithm that calculates the probability for each hypothesis then outputs the most probable.
- In K-Nearest Neighbor method, the selection of k values is tricky and application dependent. To simplify our problem, it is always fixed to odd number so that no tie can happen. In our experiment we tried to classify the credit approval testing data set for different values of k and for k=8 we got the maximum correct classification rate.

K-NN BASED BAYESIAN CLASSIFIER FOR SLOPE COLLAPSE PREDICTION -

Fig. 2 provides the overall picture of the proposed model K-NNBC which is divided into 7 steps. It is noted that the model is established based on the Bayesian inference and the K-NN approach.

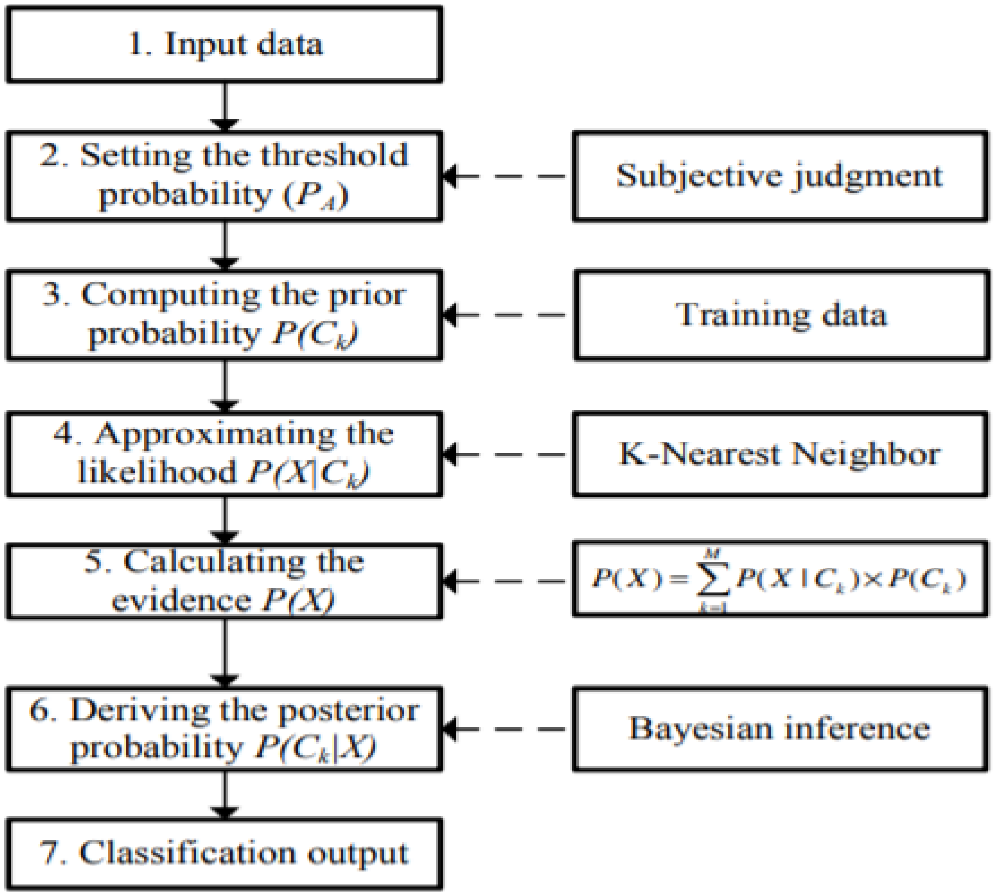


Fig. 2 *K*-Nearest Neighbor Based Bayesian Classifier

Conclusion -

This presented and verified a new prediction model, named as K-NNBC, to assist decision makers in slope collapse prediction. The proposed model is developed by the fusion of the Bayesian inference framework and the K-NN approach. The K-NNBC utilizes the Bayesian framework to compute the posterior probability of slope collapse event given an input pattern that provides features of the investigated area. Furthermore, K-NN is employed to approximate the class-conditional probability density without any assumption of the form of the density. The proposed model also does not require the assumption of independent attributes which can be delusive in the real-world situation. Superior prediction accuracy in the experiment process have convincingly proved the capability of the new approach for supporting decision-makers in slope collapse prediction as well as in disaster prevention planning.

Compare_KNN_NaivesBayes.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

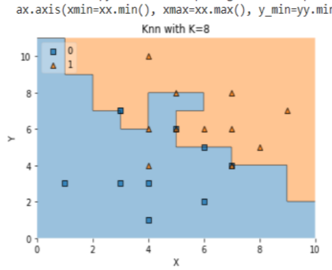
RAM Disk Editing

```
[135] print("Predict Output for k=8 is",clf.predict([[6,5]]))
```

```
Predict Output for k=8 is [1]
```

```
y_pred  =  clf.predict(x_test)
ac = accuracy_score(y_test,y_pred)
print("Using KNN the Accuracy is= ", ac)
# Plotting decision region
plot_decision_regions(x_train, y_train, clf=clf, legend=2)
# Adding axes annotations
plt.xlabel("X")
plt.ylabel("Y")
plt.title("Knn with K=8")
plt.show()
```

```
Using KNN the Accuracy is=  0.6666666666666666
/usr/local/lib/python3.7/dist-packages/mlxtend/plotting/decision_regions.py:244: MatplotlibDeprecationWarning: Passing unsupported keyword arguments to axis() will raise a TypeError ir
  ax.axis(xmin=xx.min(), xmax=xx.max(), y_min=yy.min(), y_max=yy.max())
```



Type here to search

59°F Mostly cloudy 1:47 AM 6/20/2021

Compare_KNN_NaivesBayes.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

Applying Naive Bayes Classification for same Input.

```
[137] from sklearn.preprocessing import StandardScaler
      sc = StandardScaler()
      X_train = sc.fit_transform(x_train)
      X_test = sc.transform(x_test)
```

```
[138] from sklearn.naive_bayes import GaussianNB
      classifier = GaussianNB()
      classifier.fit(X_train,y_train)
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
```

```
[139] #Predict Output
      predicted= classifier.predict([[6,5]]) # 0:+, 1:-
      print("Predicted Value:", predicted)
```

```
Predicted Value: [1]
```

```
y_pred  =  classifier.predict(X_test)
print(y_pred)
print(y_test)
cm = confusion_matrix(y_test, y_pred)
print(cm)
ac = accuracy_score(y_test,y_pred)
print("Using Bayes the Accuracy is=",ac)
```

```
[0 1 1]
[1 1 1]
[[0 0]
 [1 2]]
Using Bayes the Accuracy is= 0.6666666666666666
```

Type here to search

59°F Mostly cloudy 1:48 AM 6/20/2021